REAL-TIME MUSIC SYNTHESIS ON CROMEMCO SYSTEM-3

by

INDRANIL BASU

COMPUTER SCIENCE PROGRAM

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

AUGUST, 1982

REAL-TIME MUSIC SYNTHESIS ON CROMEMCO SYSTEM-3

A Thesis Submitted
in Partial Fulfilment of the Requirements
for the Degree of

MASTER OF TECHNOLOGY

by

INDRANIL BASU

to the

COMPUTER SCIENCE PROGRAM

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

AUGUST, 1982

CSP-1982-M-BAS-REA

CERTIFICATE

This is certify that the work entitled 'REAL-TIME MUSIC SYNTHESIS on CROMEMCO SYSTEM-3' by Indranil Basu has been carried out under my supervision and has not been submitted elsewhere for a degree.

H V Sahasrabuddhe

Dr. H.V. Sahasrabuddhe
Professor
Computer Science Program
Indian Institute of Technology
Kanpur.

Kanpur
August 1982

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my thesis supervisor, Dr. H.V. Sahasrabuddhe, for his interest, timely and valuable advice and constant encouragement. His inspiration and guidance is invaluable for this thesis to take shape.

I am also grateful to my friends Amitabha Sanyal, M. Paul and many others for making my stay at Indian Institute of Technology, Kanpur a memorable experience.

Finally, I would like to thank Mr. H.K. Nathani for typing this manuscript excellently.

- Indranil Basu

Kanpur
August 1982

CONTENTS

-

ABSTRACT

Here, we attempt to synthesize music in real-time by a digital computer, given the musical score, expressed in terms of numerical values of amplutides, frequencies and durations of sequences for each waveform for each instrument we want to represent, and also repetition clauses in between sequences where desired. We aim to achieve as high a frequency as possible entirely by software. We have been able to achieve a frequency of about 1.25KHz with sampling rates of about 5KHz.

The waveforms which we have chosen here are the sine, square and sawtooth waveforms, which are the approximations of those produced by a flute, clarinet and cello respectively. The resultant waveform, obtained after summing up the samples from each individual waveform and outputting it to the digital-to-analog converter, represents the desired musical waveform.

CHAPTER 1

INTRODUCTION

In our project, our aim was to develop a software package for music synthesis by computer. The motivation for this project lies in that here we are utilizing the immense potentialities of a digital computer for music synthesis by numerically specifying the amplitude, frequency and duration of the various waveforms to be synthesized by the computer, each wageform representing (or rather, approximating) those generated by different musical instruments.

The advantage of using a computer for synthesizing music is that a beginner, who is not familiar with the use of the various musical instruments, can have an exercise in innovative music composition, by simply specifying the numerical values of the relevant parameters, namely, amplitude, frequency, duration and for repetition clauses, where a particular tune is desired to be repeated, specifying the number of repetitions and from where to repeat. The computer computes the values of the samples, outputs them to the digital-to-analog converter, from where they are converted into sound by a loudspeaker.

The computer can be programmed to synthesize any combination of waveforms, and the waveforms too, can be of any desired type. Theoretically, any instrument waveform can be synthesized on a computer and this greatly enhances the versatility of the computer for innovative music generation, without the necessity of becoming familiar with how to use any of the musical instruments, whose waveforms can be mathematically represented and generated by the computer.

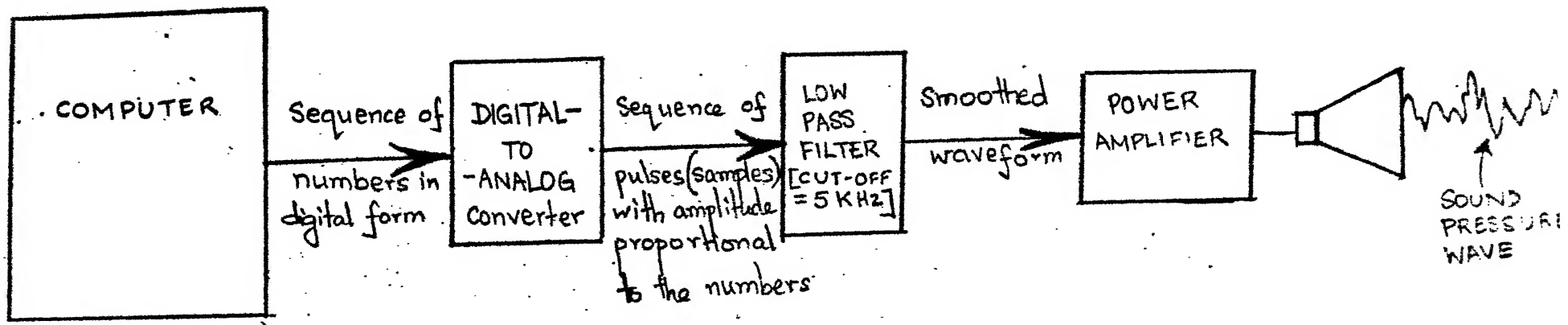

FIGURE 2.1 SCHEMATIC DIAGRAM

CHAPTER 2

FORMULATION OF THE PROBLEMS INVOLVED IN REAL-TIME SYNTHESIS OF MUSIC

2-1. Introduction

In this chapter, we discuss some fundamentals that are basic to all computer sound processing - the representation of sound as numbers, the underlying processes of sampling and quantizing a sound wave, the approximations and errors that are inherent in sampling and quantizing, the operation of digital-to-analog converters, the construction of smooting filters, and, last, an introductory look at the computer programming for music synthesis, which is the central subject of this project.

2-2. Numerical Representation of Functions of Time

Sound can be considered as a changing or time-varying pressure in the air. Its subjective characteristics, how it "sounds", depends on the specific way the pressure varies.

Since the essence of the sound depends on the nature of the variations in pressure, we will describe a sound wave by a pressure function p(t), where the pressure p is a function of time t.

All sounds have a pressure function and any sound can be produced be generating its pressure function. Thus if we can develop a pressure source capable of producing

any pressure function, it will be capable of producing any sound, including speech, music, and noice. A digital computer, plus a program, plus a digital-to-analog converter, plus a loudspeaker come close to meeting this capability.

In the past most sounds have originated from the vibrations and movements of natural objects - human and vocal cords, violin strings, collision between two solid objects, etc. The nature of these sounds is determined by and limited by the particular objects. However, in the last 60 years the loudspeaker has been developed as a general sound source. It produces a pressure function by means of the vibrations of a paper cone actuated by a coil of wire in a magnetic field. The movement of the cone as a function of time, and hence the resulting pressure function, are determined by the electric voltage (as a function of time) applied to the coil.

2-3. Sampling and Quantizing

Figure 2-1 illustrates broadly the scheme adopted here for generation of a sequence of samples. In the processor, we generate a sequence of representative numbers for the samples. Each number simply gives the values of the function at one instant, in time. Practically, the computer can produce any set of numbers and hence any function. However, some functions are more

difficult to produce than others, and certain approximations are involved in producing any function. It is important to understand the nature of these approximations in order to use the computer as an effective sound source. Sampling and quantizing are the two approximations involved in representing a continuous function by a set and numbers.

Mathematically it has been shown that R samples per second are needed to approximate perfectly a function with a bandwidth R/2 cycles per second. In our case, the maximum sampling rate for the digital-to-analog converter is 10,000 samples/sec, which implies that the maximum frequency of the sampled signal should not be more than 2,000 cycles/sec (or 2 KHz), for 5 samples per cycle. Thus, the minimum possible sampling interval is Ø.ØØØ1 sec. = Ø.1 msec.

The second approximation is called quantizing.

2-4. Foldover Errors

The sampling theorem, for sampled data signals, states that if a signal contains no frequency components above a certain value f_w, then it is completely described by instantaneous sample values of frequency greater than or equal to twice the signal frequency f_w.

The minimum sampling rate, $f_s = 2f_w$ is known as the Nyquist rate.

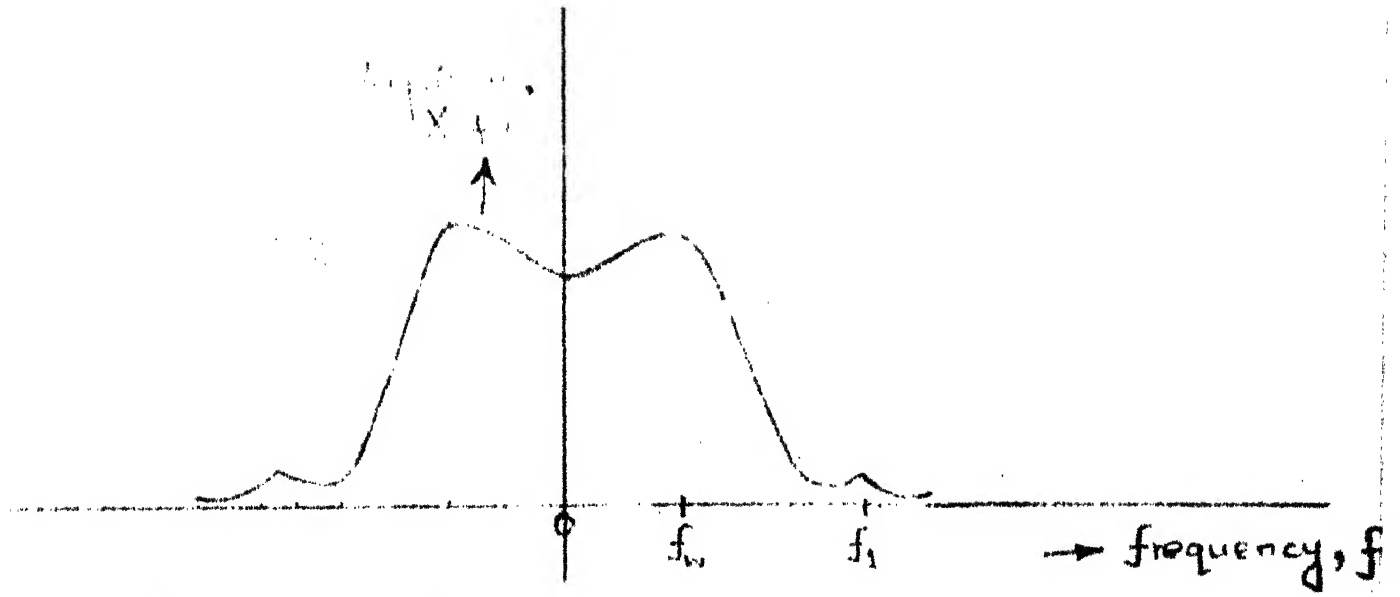

Fig. 2.2 Nonbandlimited signal spectrum

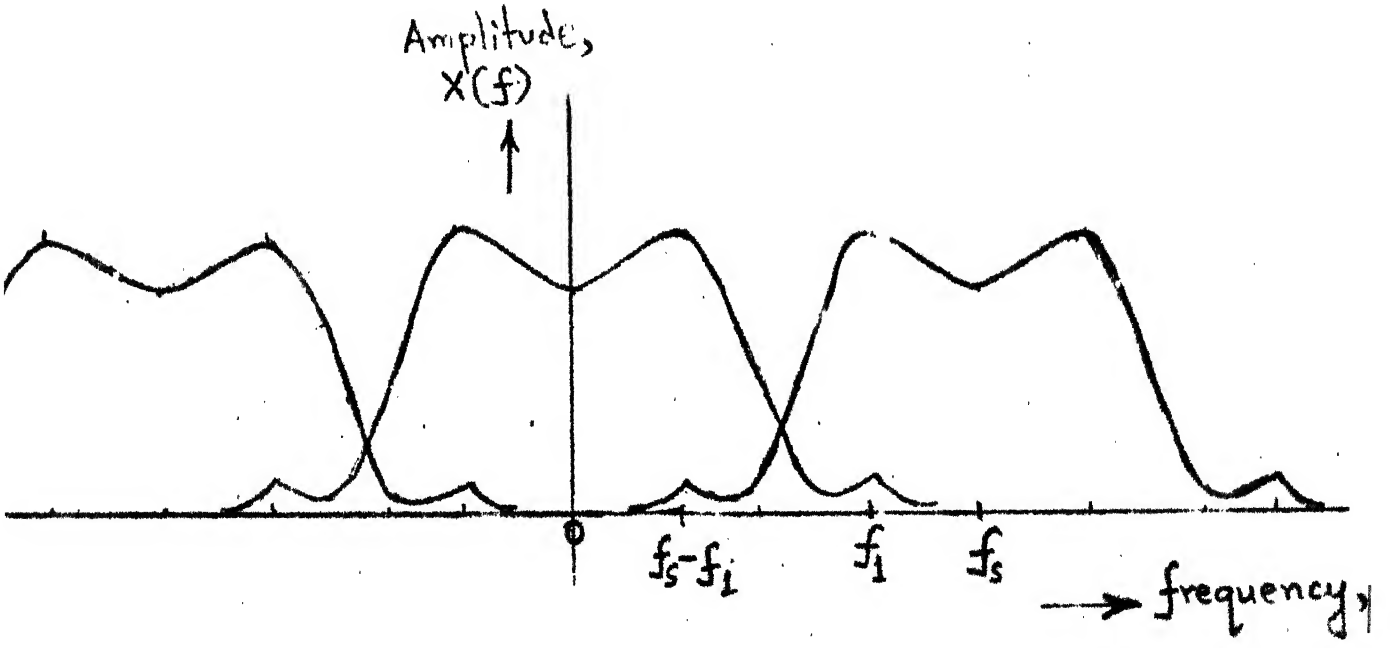

Fig. 2.3 Aliasing effect (or undersampling) due to signal frequency components greater than $f_s/2$

If a signal has been sampled at the nominal Nyquist rate or greater ($f_s \geq 2f_w$) and the samples are represented as amplitude modulated pulses, the signal can be approximately reconstructed from its samples by low-pass filtering, whose cut-off frequency = f_w.

A signal specturm, such as shown in Figure 2-2, is considered to be virtually bandlimited if the frequency content above W is small and presumably unimportant for conveying the information. When such a message is sampled, there will be unavoidable overlapping of spectral components (Figure 2-3). In reconstruction, frequencies originally outside the nominal message band will appear at the filter output on the form of much lower frequencies. Thus, for example, $f_1 > w$ becomes $f_s - f_1 < w$, as indicated in the figure.

The phenomenon of downward frequency translation occurs whenever a frequency component is undersampled, that is, $f_s < 2f_1$, and is given the descriptive name of aliasing. The aliasing effect is far more serious than spurious frequencies passed by nonideal reconstruction filters, for the latter fall outside the message band, whereas aliased components can fall within the message band. Aliasing is combated by filtering the message as much as possible before sampling and sampling at much greater than the nominal Nyquist rate, if necessary.

Here, in our system, after sampling, the sampled pulses are passed on to the digital-to-analog converter's digital input part. If the signal frequency is f_w samples occurs at a rate of $f_s \geq 2f_w$ samples per second, so there must be $8f_s$ coded pulses per second, since each sample is represented by 8-bit codes. If there are no "spaces" in the coded signal, the maximum permitted duration of any one pulse is $= 1/8f_s$. On the basis of pulse resolution, we need a bandwidth of at least $\frac{1}{2}(8f_s)$.

$$\therefore \text{Bandwidth} \geq \frac{1}{2}(8f_s) \geq 8f_w.$$

The threshold signal-to-noise ratio

$$= 10(\mu^2 - 1),$$

where $\mu = 2$ for binary digits.

The approximate signal-to-noise ratio inherent in a given number of bits equals

$$= \frac{\text{Maximum number expressible}}{\text{Maximum error in representing any number}}$$

For our case, the maximum number expressible is $2^8 - 1 = 255$, and the maximum error is 0.5

$$\therefore \text{The signal to noise ratio} = \frac{255}{0.5}$$

$$= 510$$

$$= (20 \log_{10} 510)\text{dB}$$

$$\underline{\sim} \ 54\text{dB}.$$

This ratio is as good as very high quality audio equipment, Hence, quantizing error is not a critical factor here. Sampling rate, by contrast, is often critical.

2-5. Tones and Harmonics

The elementary unit of a musical composition is the note. Its position on the staff indicates a pitch to be played at a particular time. The tonal quality of the note is not generally indicated in a musical score, since that is already determined by the instrument assigned to play the note. A composer seeking a particular tone colour obtains it by selection of the proper instrument on combination of instruments.

To program a computer to synthesize a tone, the composer must specify much more than the tone's pitch. He or she must be able to define the desired tone colour and articulation and must also provide all the accoustical information needed to produce that particular sound. To supply such information to the computer, the programmer must describe the characteristics of the tone in mathematical terms.

2-6. Digital-to-Analog Converter

Conversion between numbers in a computer and analog voltages is an essential step in sound processing. It is conceptually simple and practically easy to accomplish. A variety of commercial equipment are available. Here, a digital-to-analog converter is there as part of the Cromemco System.

Out digital-to-analog converter can have a digital input and analog output with maximum sampling rate of 10KHz.

2-7. Smoothing Low-Pass Filter

In order to smooth out the sampled analog signal at the output of the digital-to-analog converter, we need a smoothing low pass filter, with a cut-off frequency equal to the maximum signal frequency.

2-8. Fundamental Programming Problems for Sound Synthesis

Since we have described the technology for sound processing by computer, we now look at the computer programs that activate this technology.

The two fundamental problems in sound synthesis are: (1) The vast amount of data needed to specify a pressure function - hence the necessity of a very fast and efficient computer program - and (2) the need for a simple, powerful language in which to describe a complex sequence of sounds. To solve these problems we use the following principles:

(1) Stored functions to speed computation, (2) waveform generator blocks for sound-synthesizing instruments to provide great flexibility, and (3) the note concept for describing sound sequences. We will now consider sound synthesis from the computer's and the composer's stand points to see the importance of these principles.

To specify a pressure function at a sampling rate of say, 10KHz, one number is needed every 100 microseconds. A useful measure of computation is the time scale, which is defined as

$$\text{Time Scale} \equiv \frac{\text{Time to compute samples of a sound}}{\text{Duration of the sound}}$$

Various possibilities exist at various time scales. If the time scale is equal to 1 or less, a digital-to-analog converter can be attached directly to the computer and sound can be synthesized in real time. This allows improvising on the computer, hearing the sound immediately after specifying the musical score. To ensure real-time synthesis of music, the computations for each sample must be few.

Time scales greater than 1 necessitate recording the samples in secondary memory (disk or tape), rewinding the tape, and playing the tape through the digital-to-analog converter. A delay equal to or greater than the sound duration is inherent in the process. For time scales above 50, the delays become too time consuming and expensive

and not worth the effort.

We have considered sound synthesis from the position of the computer and it has led us to sampling functions. Now, we take a look from the composer's standpoint. He would like to specify any sequence of sounds in a musical score memory, and a program which will read these musical scores are play the desired musical sequence. The musical score consists of a number of tracks, each track for each musical instrument that is desired by the compower, and for each musical instrument, a subroutine subprogram for computing the values of the samples over a cycle must be written.

The final principle for specifying sound sequences is the note concept. Sound exists as a continous function of time starting of the beginning of a piece and extending to the end. We have chosen, for practical reasons, to chop this continous sound into discrete pieces, called notes, each of which has a starting time and a duration time.

In the musical score tracks, in each track, the composer specifies numerically sequences of amplitude, frequency and duration for the waveforms represented by the respective tracks. Also, if any sequence is to be repeated, then a repeatition clause is to be inserted at the required place, specifying the number of repetitions required and from where to repeat.

Thus, we have a working program for music synthesis in real-time, based on the above principles. With such a program, the user, by specifying any combinations of instrument waveforms (a subroutine has to be written for each) and any sequence of their amplitudes, frequencies, durations and repetition clauses, can really be able to do innovative music composition with the computer.

CHAPTER 3

TECHNIQUE ADOPTED BY US FOR REAL-TIME MUSIC SYNTHESIS

3-1. Introduction

Our aim in this project is to design a real-time music synthesis software package for the Cromemco-System-3 microcomputer. The music to be produced is a combination of tones produced by different musical instruments, such as flute , clarinet and cello of varying amplitudes, frequencies and durations, as specified in the musical score memory.

The musical score memory contains the musical score represented in terms of records of amplitude, frequency and duration and repetition clauses where needed, on tracks, each track for each musical instrument desired. The waveforms produced by the different musical instruments will be different and we can choose our own combination of instruments to be played, by calling subroutines to generate waveforms for those particular musical instruments.

For each instrument wavefoorm, we have a subroutine which computes the values of the sample for each instrument waveform, over a cycle within the duration specified and forms tables of their values, which are then passed on to the main program. The main program continously

sums up the sample values from each instrument waveform and outputs them to the digital-to-analog converter part. As soon as any of the durations for any waveform is over, the table for the next set of sample values for that waveform are computed and obtained from the subroutine for that waveform and the main program goes on until any of the next set of durations expires, when it again obtains the table for the next set of sample values for that waveform and resumes outputting of these values, represented by proportionate voltage pulses, to the digital-to-analog converter part. This process goes on until the total time limit specified for the music to play is over.

3-2. Description of the Hardware System

Here we have used the CROMEMCO System-3 Microcomputer on which to implement and run our program for real time music synthesis. It has a Z80A 8 bit-CPU, with a cycle time of 250 nanoseconds, a memory of 64K bytes with a maximum memory access time of 250 nanoseconds.

The digital-to-analog converter ("Cromemco D+7A I/O Module") is a fasthigh-perfirmance converter, with a fast conversion time of 5.5 microseconds.

Analog Output Port Specifications

Number of output ports
Output voltage range : -2.56volts to +2.54 volts
Output impedance : 0.25 ohms
Maximum load current : 1.5mA
Rsolution : 8 bits
Conversion time : 5.5 microseconds
Accuracy : $\pm$ 20 millivolts
Drift rate : Less than 10 mV/Sec at 25^{o}C.

3-3. Description of Our Program for Real-Time Music Synthesis on CROMEMCO System-3

In the program for real-time music synthesis on the CROMEMCO System-3, we aim to compute sample values, at the s ampling instants, of the chosen waveforms (which are approximations of these produced by instruments we want to represent). Having obtained the required sample values from the individual waveforms, we compute the resultant sample of the different instruments being played simultaneously, by summing up the individual samples. Having thus obtained the sample for the composite musical waveform, it is output to the digital-to-analog converter port. At the analog output of the digital-to-analog converter port, the sampled waveform is smoothened out by a smoothing (low-pass) filter, and then passed on to a loudspeaker, for conversion into sound waves. The resultant frequency of the output musical waveform will depend on the sampling rate, and the number of samples per cycle of the original individual waveforms. Thus, if we have at least 3 samples per cycle (for the highest frequency waveform) then by having a sampling rate of 5KHz, we can have a maximum frequency of about 1.6KHz for the output waveform. Thus, to achieve high output musi frequencies, we should aim for as high a sampling rate as possible.

In our program, for each individual waveform representing that from an instrument which we want to simulate, we have written a subroutine which computes the values of the samples for that waveform at the sampling instants, over a cycle of that waveform, within the given duration. The duration specified is that for that waveform at the given amplitude and frequency. The sample values are put in a table, which is then passed on to the main program. Thus, here in our program, we have subroutines for generating tables of sample values for sinewave, squarewave and sawtoothwave, which are approximations of the waveforms produced by flute, clarinet and cello respectively.

The music generating program reads the musical score from an input file and produces musical output accordingly. The musical score consists of a number of tracks equal to the number of waveforms we want to represent, each track for each waveform. On each track, we specify the sequence to be played out by the corresponding waveform, the sequence being represented here in the track by a series of records of three fields each. The first, second and third fields contain real number representations of certain parameters pertaining to that particular waveform. If the first field contains a positive number, it indicates that the record consists of amplitude, frequency and duration specifications in the first, second and third fields respectively. If,

on the other hand, the first field contains a negative number, it indicates that this record is a repetition clause and the magnitude of its first, second and third fields contain respectively the number of repetitions, a pointer to the location on this track from where to repeat, and an indicator variable.

In order to generate samples for the musical waveform, the main program calls all the instrument waveform generating subroutines once and determines the minimum of the three durations for the three different waveforms which we have here chosen to represent. Then, in the main program, the sum of the samples from each waveform is determined and outputted to the digital-to-analog converter port. This process computing the sum of the samples of the individual waveforms and outputting it goes on until the minimum of the three durations of the three waveforms is over. Then the subroutine for that waveform whose duration is over is called and the next set of samples for that waveform over a cycle with the duration for true new values of amplitude, frequency and duration are computed in a table, which is then passed on to the main program, which then resumes computing the sum and outputting the samples of the resultant desired musical waveform, until the minimum duration from among the present set of the three durations is over, when it calls the subroutine for which the duration is over, and so on. This process goes on until the total time limit set for the music to play is over.

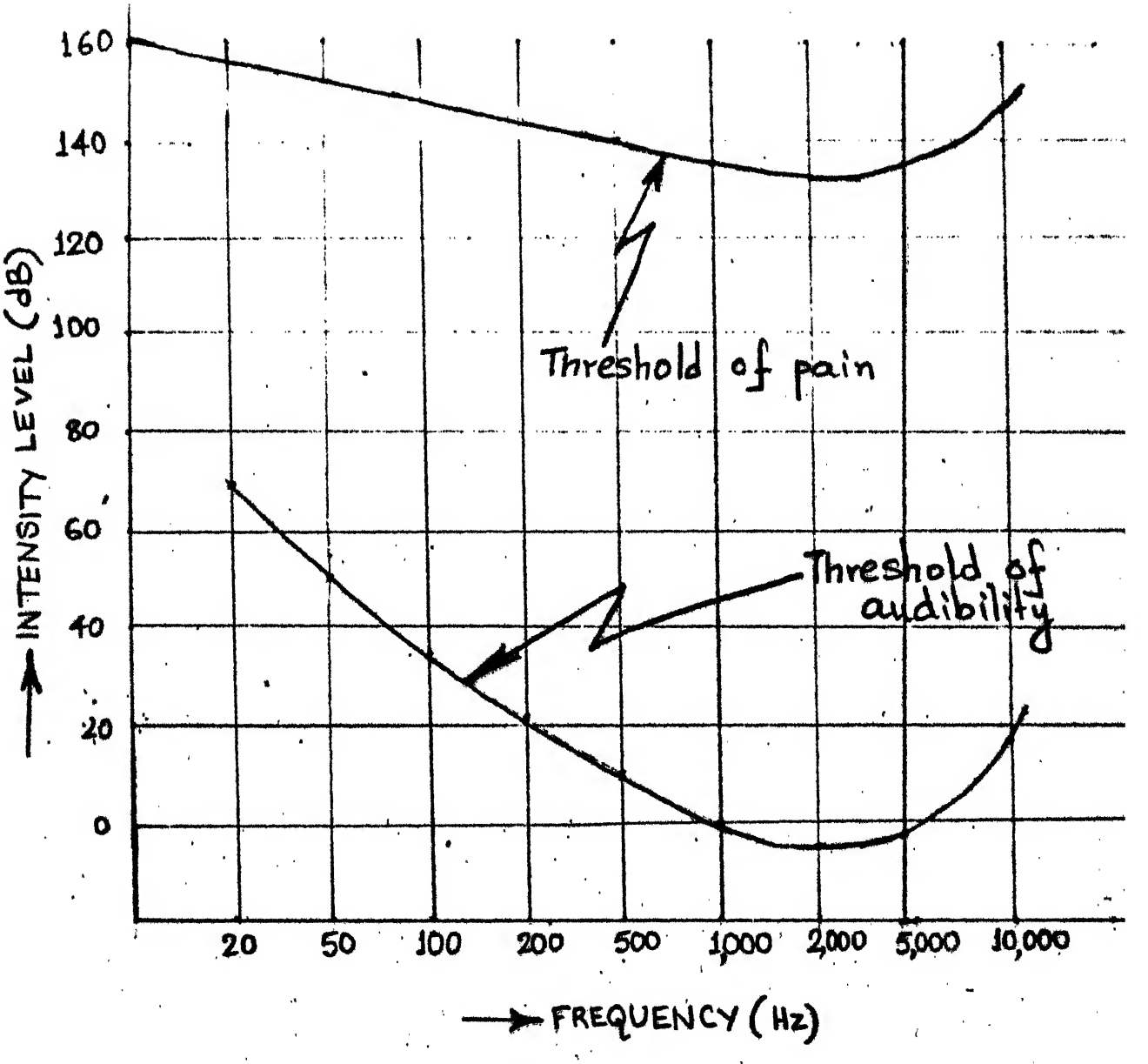

Fig. 4.1 Feeling and audibility thresholds

CHAPTER 4

PSYCHOACOUSTICS and MUSIC

4-1. Introduction

Although the technology of electronic and computer sould generation has given us new tools of almost unlimited power for making new sounds, it has also created a new problem - the need to understand the psychoacoustics of musical perception. Sounds produced by conventional instruments are so well known that composers can proceed with the intuitions they have developed from long experience. However, no such intuitions exist for new sounds. Instead, the composer must understand the relation between the physical sound wave and how it is perceived by a hearer. Psychoacoustics addresses this question and hence has become an essential knowledge for the modern composer.

With some exceptions original scientificwork in psychoacoustics has not been directed chiefly at musical problems. Thus we must draw on a variety of sources in seeking to understand musical phenomena, although investigators did not always have music in mind.

4-2. Loudness

The perceived loudness of a sound depends on many factors in addition to its intensity. For example, in order for a pure tone or sinusoid at 100Hz to be heard, its sound intensity must be 1000 times greater than that of a pure

tone at 3000Hz. For most of the musical range the perceived loudness increases as the 0.6 power of the sound pressure. The perceived loudness increases more slowly with sound pressure for 3000Hz tones than it does for very low frequencies, say, 100 Hz; and in the uncomfortably lound range, tones of equal power are about equally lound. This means that as we turn the volume control up or down, the balance of loudness among frequency components changes slightly.

4-3. Masking and Threshold Shifts

A tone or a noise masks or renders us incapable of hearing a less powerful tone. A tone has a strong masking effect for tones of higher frequency and a weaker masking effect for tones of lower frequency. The frequency range of masking is greater for lound tones than for soft tones. Thus we would expect that in a musical composition some sounds might be masked and unheared when the volume is set high, whereas they would be unmasked and heard when the volume is low.

Masking can be considered as a raising of the level at which tones become audible. Some rise in the threshold persists for 1/6 sec or longer after a lound tone but the aftereffect of a lound tone on hearing is much less than that of a bright light on seeing.

4-4. Limens or Just Noticeable Differences

Limens or jnd's of loudness and frequency have been carefully measured. They are surprisingly small. However, there is evidence that the limens are much smaller than the frequency or loudness differences that can be detected in complicated listening tasks, which are more akin to music. Very small differences in frequency (less than a half tone) and loudness can be detected in successive tones that are not too short.

4-5. Pitch

The pitch of a complex tone is often thought of as that of its lowest partial. However, experiments made with repetitions of various patterns of pulses and with complex tones in which the upper partials are harmonics of a frequency higher than the fundamental show that, although the fundamental dominates at higher frequencies, the repeition rate of the tone or of its higher partials dominates at lower frequencies. The pitch of a tone may be highly uncertain by one or more octaves; thus circle of 12 tones was produced, which when cyclically repeated give the impression of always rising in pitch, with no break. Tones with inharmonic partials, including gongs, bells, and tones specially synthesized by computers may produce a sensation of pitch; a tune can be played on them. But the pitch may not be the first partial, for example, the hum tone of a

bell is not the pitch to which the bell is tuned.

4-6. Quality or Timbre of Steady Tones

The sound quality or timbre of steady tones depends on the partials. Although partials up to the sixth can be heard individually by careful listening, we tend rather to hear an over-all effect of the partials through the timbre of the tone. A pure tone or sinusoid is thin. A combination of octave partials is bright. A tone with a large number of harmonic partials is harsh or buzzy. In general, the timbre appears to be dissonant or unpleasant if two strong partials fall within a critical bandwidth, which is about 100 Hz below 600 Hz and about a fifth of an octave above 600 Hz.

The timbre of a sound is strongly affected by reasonances in the vocal tract or in musical instruments. These resonances strengthen the partials near the resonant frequencies. Threee important formants or ranges of strengthened frequency are produced by the vocal tract; they give the qualities to vowel sounds which are identifiable independent of pitch.

4-7. Transient Phenomena

Textbooks give harmonic analyses of the sounds of various musical instruments, but if we synthesize a steady tone according to such a formula it sounds little like the actual instrument. Steady synthesized vowels do not sound like speech if their duration is long.

Temporal changes such as attack, decay, vibrato, and tremolo, whether regular or irregular, have a strong effect on sound quality. A rapid attack followed by a gradual decay gives a plucked quality to any waveform. Also, the the rate of which various partials rise with time and the difference in the relative intensity of partials with loudness are essential to the quality of the sound. Indeed it is at least in part the difference in relative intensity of partials that enables us to tell a loud passage from a soft passage regardless of the setting of the volume control. This clue is lost in electronic music if the tones employed have a constant relative strength of partials, independent of volume.

The "warmth" of the piano tone has been shown to be due to the fact that the upper partials are not quite harmonically related to the fundamental.

4-8. Consonance

Observers with normal hearing but without musical training find pairs of pure tones consonent if the frequencies are separated by more than the critical bandwidth or if the frequencies coincide or are within a few hertz of one another (in this case beats are heard). Pairs of tones are most dissonant when they are about a quarter of a critical bandwidth apart. For frequencies above 600 Hz, this is about a twentieth of an octave.

Excluding bells, gongs, and drums, the partial of musical instruments are nearly harmonic. When this is so, for certain ratios of the frequencies of fundamentals, the partials of two tones either coincide or are well separated. These ratios of fundamentals are 2:1 (the octave), 3:2 (the fifth, 4:3 (the fourth), 5:4 (the major third), and 6:5 (the munor third). Normal observers find pairs of tones with these ratios of fundamentals to be more pleasant, and intervening ratios less pleasant.

Musical consonance and dissonance depend on many factors in addition to frequencies of partils. For example, unlike nonmusicians, classically trained musicians describe pairs of pure tones with these simple numerical ratios of frequency as consonant and intervening ratios as dissoant. The only reasonable explanation is that trained musicians are able to recognize familiar intervals and have learned to think of these intervals only as consonant.

It has been pointed out that, in order for complex tones to attain a given degree of consonance, low tones must be separated by a larger fraction of an octave than high tones, and it was observed that composers follow this principle.

If the partials of a tone are regularly arranged but not harmonic, the ratios of frequencies of the fundamental (or first partial) that lead to consonance are not the conventional ones.

4-9. Combination Tones

When we listen to a pure tone of frequency f_1 and another tone of somewhat higher frequency f_2, we hear a combination tone of lower frequency $2f_1 - f_2$, even at low sound levels. At much higher sound levels, around 100,000 times or more the power at threshold, it is possible to hear faint frequencies $2f_1$, $2f_2$, $f_1 + f_2$, $f_2 - f_1$, etc. Combination tones are due to nonlinearities in the hearing mechanism. They can contribute to dissonance and to beats.

4-10. Reverberation

Reverberation is important to musical quality; music recorded in an organ loft sounds like a bad electronic organ. The reverberation for speech should be as short as possible; for music about 2 sec is effective. Music sounds dry in a hall designed for speech. Reverberation is not the only effect in architectural acoustics. Our understanding of architectural acoustics is far from satisfactory.

4-11. The Choir Effect

Many voices or many instruments do not sound like one voice or one instrument. Some experiments by Mathews show that a choir effect cannot be attained by random tremolo

or vibrato. It must be due to irregular changes in overall waveform, caused by beating or head motions, or by differences in attach.

4-12. Direction and Distance

We can experience a sideness to sound by wearing headphones fed from two microphones, but the sound seems to be inside our head. We experience extrernalization of the sound - as coming from a particular direction - only when we allow head movements in a sound field. Although we cannot detect the direction of the source of a sinusoidal tone in a reverberant room, we can detect the direction by the onset of such a tone, and we can detect the direction of clicks and other changing sounds. The first arrival of the sound dominates later reverberant arrivals in our sensing of the direction of the source; this is called the precedence effect. We can detect vertical angle of arrival although no one is sure how this is done. We can also sense the distance of a source in a reverberant room; this sensation must depend on some comparison of the direct arrival and the reverberant sound.

4-13. Memory and Overlearning

Most memory experiments are not done with musical sounds, but many are relevant to music.

It has been found that subjects can remember a sequence of from 5 to 9 randomly chosen digits, letters, or words.

On the other hand, a good bridge player can remember every card that has been played in an entire game. Our ability to deal with stimuli depends on their familiarity or 'meaning' to us. This familiarity comes about through overlearning. Overlearning has been insufficiently investigated because although it is common in life, it is very difficult to achieve in the laboratory.

The phonemes of a language are overlearned. A subject can readily distinguish the phonemes of his own tongue, but not those of another. He can distinguish dialects of his own language, but not those of a foreign tongue. He can understand his native language in a noisy place better than he can understand a foreign language even though he is expert in it.

Conventional elements and structures in music are undoubtedly overlearned. Much of our appreciation of harmony, much of our ability to remember conventional tunes must depend on overlearning, just as our ability to use and remember language does. Performance with unfamiliar material is much power.

4-14. Psychological Distance; Scaling

Some psychological stimuli have the same pattern of similarity for all people. Colour is one. The psychological distance between stimuli such as colours can be obtained by computer analysis of data expressing either the confusions

that subjects make among pairs of stimuli or the numbers that they assign to the pairs to express their judgements of similarity. This kind of analysis is called multi-dimensional scaling. The stimuli may appear in a psychological space of one dimension (loudness does), two dimensions (colour does) or three (vowels do) or more dimensions. Psychological distance is dependent on, but not proportional to, physical parameters. Thus red and violet light are of all colours the farthest apart in wavelength, and yet they look more alike - they are closer together psychologically - than the "intermediate" colours orange and blue.

CHAPTER 5

CONCLUSION

In this project, we have seen how musical waveforms are generated by means of a program, which calls the subroutines for the different individual instruments and then sums up and outputs the values of the samples.

The time to compute a single sample of the resulting music waveform has been tried to be made as low as possible, by doing only the summing and outputting to the digital-to-analog converter port, in the routine which computes the samples and outputs them successively. This time has been found to be of the order of 0.2 milliseconds, for each sample. Hence, if we sample waveforms at the rate of 4 samples per cycle of the waveform, the waveform time period will come out to be of the order of 0.8 milliseconds, and hence, the frequency of this resultant musical waveform will be of the order of 1.25KHz.

Out digital-to-analog converter can convert digital signals to analog signals for sampling rates upto 10KHz, which implies, that even if we sample at the Nyquist rate of 2 samples per cycle of the signal waveform, the resulting output musical waveform can have a maximum frequency of 5KHz.

One problem encountered in our program is that, in between two successive durations of the two specification records, there is a silence of about 20 milliseconds, while the program fetches the subroutine for that waveform, whose earlier duration just expired. The subroutine then forms a table of values for the waveform over a cycle of itself, within its specified duration, and passes on this table to the main program, which again resumes outputting of the summed up samples for the resulting musical waveform. In order to avoid such long periods of silent gaps in between musical sequences, we can distribute this task of computing the next table of sample values for each instrument waveform, over the entire present duration of the music, at the cost of reducing the frequency of the output music waveform. This can be done by employing the technique of foreground-background scheduling, whereby the job which does the summing and outputting of the resultant samples is taken as the foreground (higher priority) job, while the computing of the next sets of tables for each instrument waveform is taken as the background (lower-priority) job.

To make the sampling rate faster than what we have achieved by software, we should go in for hardware, such as a hardware summer instead of summing up the individual samples in the program itself, and if possible, hardware oscillators and waveform generators, to be controlled by a master software running on a centralized processor.

BIBLOGRAPHY

1. Alles, H.G., "A Portable Digital Sound Synthesis System" Computer Music Journal 1(4), 5-9 (1977).

2. Alonso, S., J. Appleton, & C. Jones, "A Special Purpose Digital System for Musical Instruction, Composition and Performance", CHUM 10, 209-215 (1976).

3. Appleton, J., & R.C. Perera, Eds., The Development and Practice of Electronic Music, Prentice-Hall, 1975.

4. Byrd, Donald, "An Integrated Computer Music Software System", Computer Music Journal 1(2), 55-60 (1977).

5. Divilbliss, J.L., "The Real-Time Generation of Music with a Digital Computer", Journal of Music Theory 8,99-111 (1964).

6. Federkow, G., W. Buxton, & K.C. Smith, "A Computer-Controlled Sound Distribution System for the Performance of Electro-acoustic Music", Computer Music Journal 2(3), 33-42(1978).

7. Ferretti, Ercolino, "The Computer as a Tool for the Creative Musician", Computers for the Humanities? A Record of the Conference Sponsored by Yale University, Jan. 22-23, 1965, 107-112, Yale University Press.

8. Ferretti, Ercolino, "Some Research Notes on Music with the Computer", American Society of University Computers, Proceedings 1, 38-41 (1966).

9. Franco, S., Hardware Design of a Real-Time Musical System", Ph.D. thesis, University of Illinois, Urbana, 1974.

10. Freedman, David M., "A Digital Computer for the Electronic Music Studio", Journal of the Audio Engineering Society, 15, 43-50 (1967).

11. Gabura, J. & G., Ciamaga, "Digital Computer Control of Sound-Generating Apparatus for the Production of Electronic Music," Electronic Music Review, 1, 54-57 (1967).

12. Gabura, J. & G. Ciamaga, "Computer Control of Sound Appratus for Electronic Music," Journal of the Audio Engineering Society 16(1968).

13. Howe, Hubert, Electronic Music Synthesis : Concepts, Facilities, Techniques, Norton, 1975.

14. Howe, Hubert, "Composing by Computer", CHUM 9(6), 281-290, (1975).

15. Howe, Hubert, "Electronic Music and Microcomputers," Perspectives of New Music, 16(1), 70-84 (1977).

16. Koenig, Gottfried, "The Use of Computer Programmes in Creating Music", Music and Technology, Stockholm Meeting, organised by UNESCO, Jan. 1970.

17. Koenig, Gottfried, "Notes on the Computer in Music," American Society of University Composers: Proceedings 5, p. 111 (1972).

18. Lawson, J. & M. Mathews, "Computer Programs to Control a Digital Real-Time Sound Synthesizer", Computer Music Journal, 1(4), 16-21 (1977).

19. LeBrun, Marc, "Notes on Microcomputer Music," Computer Music Journal 1(2), 30-35 (1977).

20. Lincoln, H.B., Ed., The Computer and Music, Cornell University Press, 1970.

21. Manthey, Michael, "The Egg : A Purely Digital Real-Time Polyphonic Sound Synthesizer", Computer Music Journal 2(2), 32-37 (1978).

22. Mathews, Max, The Technology of Computer Music, MIT Press 1969.

23. Mathews, M. et al., "Computers and Future Music", Science 183, 263-268, 25 Jan. (1974).

24. Melby, J.B., Jr., "Some Recent Developments in Computer Synthesized Music", American Society of University Composers, Proceedings 5, p. 111 (1972).

25. Moorer, James A., "Music and Computer Composition", Communications of the Association for Computing Machinery 15, 104-113(1972).

26. Moorer, James, A., "How Does a Computer Make Music?" Computer Music Journal 2(1), 32-37 (1978).

27. Pierce, J.R., "The Computer as a Musical Instrument", Journal of the Audio-Engineering Society 8, p.139(1960).

28. Pierce, J.R., "Computer and Music", New Scientist 25, p. 423 (1965).

29. Seasy, Albert, "The Composer of Music and the Computer," Computers and Automation 13, p. (16(1964).

30. Slawson, A. Wayne, "A Speech-Oriented Synthesizer of Computer Music," Journal of Music Theory, 13,94-127 (1969).

31. Smoliar, Stephen W., "Basic Research in Computer Music Studies," Interface 2,121-125 (1973).

32. Strang, Gerald, "The Computer in Musical Composition," Computers and Automation 15, p. 16, 1966.

33. Tenney, James C., "Sound Generation by Means of a Digital Computer", Journal of Music Theory 7, 24-70 (1963).

34. Truax, Barry, "Computer Music in Canada, "NUMUS-W8, 17-26 (1975).

35. von Forster, H. and J. Beauchamp, Eds., Music by Computer, Wiley, 1969.

36. Wiggen, Knut, "The Musical Background of Computer Music," Fylkingen Internal Bulletin 2 (1969).

```
  PROGRAM FOR MUSIC SYNTHESIS USING CROMEMCO SYSTEM-3 MICROCOMPUTER
      INTEGER X,Y,Z,I,K,L,M,N,J1,J2,J3,MAXJ1,MAXJ2,MAXJ3,II1,II2,II3,
     1          IRATIO
      INTEGER*1 V
      REAL CNT,MINDUR,QNTY,
     1       TOTLTM,AMP1,AMP2,AMP3,FREQ1,FREQ2,FREQ3,
     2       DURA1,DURA2,DURA3
C*************************************************************************
C  P,Q,R ARE ARRAYS WHICH REPRESENT RESPECTIVELY THE SINE,SQUARE,AND
C  SAWTOOTH WAVE MUSICALSCORE TRACKS.EACH TRACK CONSISTS OF RECORDS
C  OF THREE FIELDS:IF THE FIRST FIELD IS A POSITIVE NUMBER IT REPRESENTS
C  THE AMPLITUDE,THE SECOND FIELD REPRESENTS THE FREQUENCY AND THE THIRD
C  FIELD REPRESENTS THE DURATION FOR THAT PARTICULAR RECORD.IF,ON THE OTHER
C  HAND,THE FIRST FIELD IS A NEGATIVE NUMBER THEN IT STANDS FOR A REPETITION
C  CLAUSE:THE MAGNITUDE OF THE FIRST FIELD THEN STANDS FOR THE NUMBER OF
C  REPETITIONS,THE SECOND FIELD CONTAINS A POINTER TO THE LOCATION OF THE
C  RECORD ON THIS TRACK TO WHICH TO GO BACK TO,WHILE THE THIRD FIELD IS AN
C  INDICATOR VARIABLE.
C                  ***********************************



      DIMENSION P(1000),Q(1000),R(1000),V1(10),V2(10),V3(10)
      DIMENSION NOREP1(100),NOREP2(100),NOREP3(100)
      DIMENSION IV1(100),IV2(100),IV3(100)
C*************************************************************************

      PI=3.1417
      DO 21 I=1,100
      NOREP1(I)=0
      NOREP2(I)=0
1     NOREP3(I)=0
      DO 32 I=1,10
      V1(I)=0.0
      V2(I)=0.0
2     V3(I)=0.0
C*************************************************************************
C  HERE THE VALUES OF N,THE LENGTH OF EACH MUSICAL SCORE TRACK,AND TOTLTM, THE
C  PROPORTIONATE VALUE OF THE TOTAL TIME FOR WHICH THE MUSIC IS TO BE PLAYED
C  ARE READ IN
C           ******************************************
      READ(1,2)N,TOTLTM
      FORMAT(1X,I3,F8.2)
C*************************************************************************
```

```
C    PROGRAM FOR MUSIC SYNTHESIS USING CROMEMCO SYSTEM-3 MICROCOMPUTER
         INTEGER X,Y,Z,I,K,L,M,N,J1,J2,J3,MAXJ1,MAXJ2,MAXJ3,II1,II2,II3,
     1           IRATIO
         INTEGER*1 V
         REAL CNT,MINDUR,QNTY,
     1        TOTLTM,AMP1,AMP2,AMP3,FREQ1,FREQ2,FREQ3,
     2        DURA1,DURA2,DURA3
C*************************************************************************
C  P,Q,R ARE ARRAYS WHICH REPRESENT RESPECTIVELY THE SINE,SQUARE,AND
C  SAWTOOTH WAVE MUSICALSCORE TRACKS.EACH TRACK CONSISTS OF RECORDS
C  OF THREE FIELDS:IF THE FIRST FIELD IS A POSITIVE NUMBER IT REPRESENTS
C  THE AMPLITUDE,THE SECOND FIELD REPRESENTS THE FREQUENCY AND THE THIRD
C  FIELD REPRESENTS THE DURATION FOR THAT PARTICULAR RECORD.IF,ON THE OTHER
C  HAND,THE FIRST FIELD IS A NEGATIVE NUMBER THEN IT STANDS FOR A REPETITION
C  CLAUSE:THE MAGNITUDE OF THE FIRST FIELD THEN STANDS FOR THE NUMBER OF
C  REPETITIONS,THE SECOND FIELD CONTAINS A POINTER TO THE LOCATION OF THE
C  RECORD ON THIS TRACK TO WHICH TO GO BACK TO,WHILE THE THIRD FIELD IS AN
C  INDICATOR VARIABLE.
C                  ************************************
C
C
C
         DIMENSION P(1000),Q(1000),R(1000),V1(10),V2(10),V3(10)
         DIMENSION NOREP1(100),NOREP2(100),NOREP3(100)
         DIMENSION IV1(100),IV2(100),IV3(100)
C*************************************************************************
C
         PI=3.1417
         DO 21 I=1,100
         NOREP1(I)=0
         NOREP2(I)=0
21       NOREP3(I)=0
         DO 32 I=1,10
         V1(I)=0.0
         V2(I)=0.0
32       V3(I)=0.0
C ************************************************************************
C  HERE THE VALUES OF N,THE LENGTH OF EACH MUSICAL SCORE TRACK,AND TOTLTM, THE
C  PROPORTIONATE VALUE OF THE TOTAL TIME FOR WHICH THE MUSIC IS TO BE PLAYED
C  ARE READ IN
C              ******************************************
         READ(1,2)N,TOTLTM
2        FORMAT(1X,I3,F8.2)
C ************************************************************************
```

```
C ***************************************************************************
C  HERE THE VALUES FOR EACH MUSICAL SCORE TRACK,THE FIRST TRACK REPRESENTING
C  SINE WAVE,THE SECOND TRACK REPRESENTING A SQUARE WAVE AND THE THIRD TRACK
C  REPRESENTING A SAWTOOTH WAVE.SINE WAVE IS AN APPROXIMATION OF THAT PRODUCED
C  BY A FLUTE,A SQUARE WAVE IS AN APPROXIMATION OF THAT PRODUCED BY A CLARINET
C  WHILE SAWTOOTH WAVE IS AN APPROXIMATION OF THAT PRODUCED BY A CELLO
C                    ****************************************
C
C
                DO 5 I=1,N
5       READ(1,11) P(I),Q(I),R(I)
C ***************************************************************************
C   HERE THE VARIOUS VARIABLES ARE INITIALIZED
C                          ==============================
C
        K=0
        L=0
        M=0
        CNT=0.0
        X=-2
        Y=-2
        Z=-2
        J1=1
        J2=1
        J3=1
        V=0
        AMP1=0.0
        AMP2=0.0
        AMP3=0.0
        FREQ1=0.0
        FREQ2=0.0
        FREQ3=0.0
        DURA1=0.0
        DURA2=0.0
        DURA3=0.0
C ***************************************************************************
```

```
C ****************************************************************************
        CALL SINTAB(P,X,NOREP1,K,V1,AMP1,FREQ1,DURA1,MAXJ1,J1)
        CALL SQRTAB(Q,Y,NOREP2,L,V2,AMP2,FREQ2,DURA2,MAXJ2,J2)
        CALL SAWTAB(R,Z,NOREP3,M,V3,AMP3,FREQ3,DURA3,MAXJ3,J3)
C ****************************************************************************
C  FINDING THE MINIMUM OF THE THREE DURATIONS FROM THE THREE TRACKS
C                    ==============================
C
52      MINDUR=AMIN1(DURA1,DURA2,DURA3)
        IMNDUR=IFIX(MINDUR)
C ****************************************************************************
        QNTY=127.0/(AMP1+AMP2+AMP3)
        DO 66 II=1,MAXJ1
66      IV1(II)=IFIX(V1(II)*QNTY)
        DO 666 II=1,MAXJ2
666     IV2(II)=IFIX(V2(II)*QNTY)
        DO 6666 II=1,MAXJ3
6666    IV3(II)=IFIX(V3(II)*QNTY)
C
C
C
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$**
C    THE ROUTINE ,WHICH DOES THE SUMMING OF THE THREE INSTANTANEOUS
C    SAMPLE VALUES AND OUTPUTS THE SUMMED AND NORMALIZED VALUE TO THE DIGITAL-
C    TO-ANALOG CONVERTER PORT
C                           **************************
C
C
        DO 6 NN=1,50
        DO 6 NNN=1,IMNDUR
        V=IV1(J1)+IV2(J2)+IV3(J3)
        CALL OUT(31,V)
        J1=J1+1
        IF(J1.GT.MAXJ1) J1=1
        J2=J2+1
        IF(J2.GT.MAXJ2) J2=1
        J3=J3+1
6       IF(J3.GT.MAXJ3) J3=1
C $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

```
C ***************************************************************************
C   HERE THE TIME COUNT SCALE IS INCREMENTED BY AN AMOUNT WHICH IS EQUAL
C   TO THE MINIMUM OF THE PRESENT DURATIONS OF THE TUNES
C                        ==========================
C
        CNT=CNT+MINDUR
C               ************************
C   HERE THE SUBROUTINE CORRESPONDING TO THE TUNE WHOSE PRESENT DURATION
C   HAS JUST EXPIRED IS CALLED TO SUPPLY THE NEXT SET OF SAMPLE VALUES
C   FOR THE NEXT SPECIFICATION RECORD
C                     ----------------------
        IF(MINDUR.EQ.DURA1) GO TO 55
        DURA1=DURA1-MINDUR
        GO TO 8
55      CALL SINTAB(P,X,NOREP1,K,V1,AMP1,FREQ1,DURA1,MAXJ1,J1)
8       IF(MINDUR.EQ.DURA2) GO TO 555
        DURA2=DURA2-MINDUR
        GO TO 88
555     CALL SQRTAB(Q,Y,NOREP2,L,V2,AMP2,FREQ2,DURA2,MAXJ2,J2)
88      IF(MINDUR.EQ.DURA3) GO TO 5555
        DURA3=DURA3-MINDUR
        GO TO 888
5555    CALL SAWTAB(R,Z,NOREP3,M,V3,AMP3,FREQ3,DURA3,MAXJ3,J3)
888     MINDUR=AMIN1(DURA1,DURA2,DURA3)
C               ****************************
        IF(CNT.LT.TOTLTM) GO TO 52
11      FORMAT(3F8.2)
        STOP
        END
C***************************************************************************
```

```
C*************************************************************************************
C   THIS SUBROUTINE COMPUTES THE SAMPLES OF THE SINE WAVE OVER A CYCLE, WITHIN
C   THE GIVEN DURATION AND FORMS A TABLE OF VALUES OF THE SAMPLES
C                ======================================
C
      SUBROUTINE SINTAB(RP,IX,INREP1,IK,RV1,A1,F1,D1,IMAXI1,II1)
      DIMENSION RP(1000),INREP1(100),RV1(20)
      PI=3.1417
      I1=0
      IX=IX+3
      RCNT1=0.0
717   I1=I1+1
10    A1=RP(IX)
      F1=RP(IX+1)
      D1=RP(IX+2)
      T1=1/(F1)
C     **********************
C   HERE THE SIGN OF A1 IS CHECKED TO SEE WHETHER THE RECORD IS
C   A TUNE SPECIFICATION OR A REPETITION CLAUSE
C                ----------------------
      IF(A1.GE.0) GO TO 125
C     ***********************
      IF(D1.NE.0) GO TO 130
      IK=IK+1
      INREP1(IK)=-IFIX(A1)
      RP(IX+2)=1
130   INREP1(IK)=INREP1(IK)-1
      IF(INREP1(IK).LT.0) GO TO 150
      IX=IFIX(RP(IX+1))
      GO TO 10
150   RP(IX+2)=0.0
      IX=IX+3
      IK=IK-1
125   IRG=IFIX(F1*RCNT1)
      RRG=FLOAT(IRG)
      ARG=(F1*RCNT1)-RRG
      RV1(I1)=A1*SIN(ARG)
133   RCNT1=RCNT1+0.02
      IF(RCNT1.GT.D1) GO TO 100
      IF(RCNT1.LE.T1) GO TO 717
100   IMAXI1=I1
      II1=1
      RETURN
      END
C@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
```

```
C&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
C   THIS SUBROUTINE COMPUTES THE VALUES OF THE SQUARE WAVE OVER A CYCLE
C   WITHIN THE GIVEN DURATION AND FORMS A TABLE OF VALUES OF THE SAMPLES
C                   ===================================
      SUBROUTINE SQRTAB(RQ,IY,INREP2,IL,RV2,A2,F2,D2,IMAXI2,II2)
      REAL T2,HLFPRD
      DIMENSION RQ(1000),INREP2(100),RV2(20)
      I2=0
      RCNT2=0.0
      IY=IY+3
818   I2=I2+1
20    A2=RQ(IY)
      F2=RQ(IY+1)
      D2=RQ(IY+2)
      T2=1/(F2)
      HLFPRD=(T2)/2.0
C     ********************************
C   HERE THE SIGN OF A2 IS CHECKED TO SEE WHETHER THE RECORD IS
C   A TUNE SPECIFICATION OR A REPETITION CLAUSE
C             ----------------
      IF(A2.GE.0) GO TO 225
C     ********************************
      IF(D2.NE.0) GO TO 230
      IL=IL+1
      INREP2(IL)=-IFIX(A2)
      RQ(IY+2)=1.0
230   INREP2(IL)=INREP2(IL)-1
      IF(INREP2(IL).LT.0) GO TO 250
      IY=IFIX(RQ(IY+1))
      GO TO 20
250   RQ(IY+2)=0.0
      IY=IY+3
      IL=IL-1
225   CONTINUE
      IF(RCNT2.GT.HLFPRD) GO TO 260
      RV2(I2)=A2
      GO TO 270
260   IF(RCNT2.GE.T2) GO TO 265
      RV2(I2)=-A2
      GO TO 270
270   CONTINUE
233   RCNT2=RCNT2+0.02
      IF(RCNT2.GT.D2) GO TO 265
      IF(RCNT2.GT.T2) GO TO 265
      GO TO 818
265   IMAXI2=I2
808   II2=1
      RETURN
      END
C&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
```

```
C#####################################################################
C  THIS SUBROUTINE COMPUTES THE VALUES OF THE SAMPLES OF THE SAWTOOTHWAVE
C  OVER THE GIVEN CYCLE WITHIN THE GIVEN DURATION AND FORMS A TABLE OF
C  THESE VALUES
C            ==========================================
C
C
      SUBROUTINE SAWTAB(RR, IZ, INREP3, IM, RV3, A3, F3, D3, IMAXI3, II3)
      DIMENSION RR(1000), INREP3(100), RV3(20)
      I3=0
      RCNT3=0.0
      IZ=IZ+3
919   I3=I3+1
30    A3=RR(IZ)
      F3=RR(IZ+1)
      D3=RR(IZ+2)
      T3=1/(F3)
C     ****************************
C  HERE THE SIGN OF A3 IS CHECKED TO SEE WHETHER THE RECORD IS
C  A TUNE SPECIFICATION OR A REPETITION CLAUSE
C     ----------------------------
      IF(A3.GE.0) GO TO 325
C     ****************************
      IF(D3.NE.0) GO TO 330
      IM=IM+1
      INREP3(IM)=-IFIX(A3)
      RR(IZ+2)=1.0
330   INREP3(IM)=INREP3(IM)-1
      IF(INREP3(IM).LT.0) GO TO 350
      IZ=IFIX(RR(IZ+1))
      GO TO 30
350   RR(IZ+2)=0.0
      IZ=IZ+3
      IM=IM-1
325   CONTINUE
      IF(RCNT3.GT.T3) GO TO 360
      RV3(I3)=A3*(1-2.0*(RCNT3/T3))
      GO TO 370
370   CONTINUE
333   RCNT3=RCNT3+0.02
      IF(RCNT3.GT.D3) GO TO 360
      IF(RCNT3.GT.T3) GO TO 360
      GO TO 919
360   IMAXI3=I3
909   II3=1
      RETURN
```